Uživatelské datové typy – výčet hodnot a interval

Typem v jazyku Delphi rozumíme atribut datových objektů, který specifikuje jednoznačně množinu hodnot a množinu operací. Seznámili jsme se již s některými standardními typy definovanými v jazyce Deplhi jako Integer, Real, Boolean, Char. U těchto typů jsou množiny hodnot i množiny operací definovány jazykem. Pro účely definic typů je v jazyce Delphi zavedeno několik abstraktních datových typů, jimiž jsou specifikovány pouze množiny operací a nikoliv množiny hodnot. V rámci těchto abstraktních typů se v jazyku Delphi definují konkrétní typy, tak že se podle určitých pravidel definují konkrétní množiny hodnot příslušející těmto datovým typům.

Výčet některých abstraktních typů, kterými se budeme zabývat:

- Výčtový typ
- Interval
- Strukturované typy:
- Množina
- Pole
- Záznam
- Soubor
- Ukazatel

Pro každý abstraktní typ je zavedena konstrukce, pomocí níž se definuje konkrétní typ a hodnoty tohoto typu. Popis konkrétního typu se může vyskytovat jako označení typu v deklaraci proměnné (nepojmenované typy), častěji však nový typ definujeme společně s jeho jménem tj. s novým identifikátorem typu. Pojmenované typy se zavádějí v úseku deklarací typů.

Příklad deklarace typů:

```
type
  Int = 1..20;
  Den=(pondeli,utery,streda);
```

Za úsekem deklarace typů mohou následovat deklarace proměnných.

```
var     A : Den;
      Dny : Den;
```

Pro deklarace platí pravidlo, že každý abstraktní datový typ musí být nejdříve deklarován a teprve poté může být použit v deklaraci proměnné!!! Na závěr je třeba uvést, že termín typ budeme používat ve dvou spojeních s různým významem. Ve spojení s identifikátorem typu, např. Integer, čímž označujeme konkrétní datový typ a ve spojení s abstraktním typem, např. pole, čímž označujeme celou třídu konkrétních typů definovaných v programu. Čili řekneme–li, že nějaká proměnná je typu pole, znamená to, že nejdříve musí být definován konkrétní datový typ pole s identifikátorem pole a s vymezenými přípustnými hodnotami pole.
V následujících kapitolách probereme jednotlivé datové typy.

Výčtový typ

Při psaní programu se často setkáváme s potřebou zobrazit určitý údaj, který nabývá pouze malého počtu hodnot, které nejsou obsaženy v žádném ze standardních typů. Konkrétní výčtový typ se definuje seznamem identifikátorů reprezentujících hodnoty tohoto typu. Výčtové typy zavádíme vždy jako pojmenované typy v úseku deklarací typů.

Příklad:
type TDen = (pondeli, utery, streda, čtvrtek, patek, sobota, nedele);

var Dny : TDen;

Každý výčtový typ je ordinální a platí pro něj standardní funkce succ, pred a ord. Uspořádání hodnot výčtového typu je dáno pořadím jednotlivých identifikátorů v definičním seznamu (první identifikátor označuje nejmenší hodnotu). Nejmenší hodnota má ordinální číslo 0, ordinální čísla ostatních hodnot se zvětšují o 1.

Příklad:
succ (pondeli) = utery, ord (pondeli) = 0, před (utery) = pondeli.
Pro výčtové typy nejsou definovány kromě relací žádné další operace.

Typ interval

Interval je typ, který specifikuje neprázdnou souvislou podmnožinu hodnot nějakého ordinálního typu. Dolní a horní mez této podmnožiny specifikují dvě konstanty daného ordinálního typu. Každý interval může být zaveden tak, že je pojmenován v deklaraci typu, nebo je deklarován bez pojmenování typu přímo jako typ proměnné.
Příklad definice datového typu s pojmenováním typu :

type Int = 1..100;
var A : Int;

Příklad deklarace proměnné typu interval bez pojmenovaného typu:

var A : 1..100;

Každý interval je typ, který je kompatibilní se svým hostitelským typem. Tento vztah znamená, že pro objekty typu interval jsou definovány tytéž operace jako pro objekty hostitelského typu a že tedy objekty typu interval mohou být použity při konstrukci výrazů tak, jako by to byly objekty hostitelského typu. Stanovením intervalu jako typu proměnné se tedy neomezuje množina operací použitelných pro tuto proměnnou, ale pouze množina hodnot.